Evolving Infrastructure for Avionics Verification and Validation Activities

Francesco Pace[(1)], Valentín Barrena[(1)], Niklas Lindman[(2)], Quirien Wijnands[(2)]

[(1)]GMV A.D.
Calle Isaac Newton 11, 28760 Tres Cantos, (Madrid)
Spain
Email: fpace@gmv.com
Email: vbarrena@gmv.com

[(2)]ESA-ESTEC
Keplerlaan 1, 2201 AZ Noordwijk ZH,
The Netherlands
Email: Niklas.Lindman@esa.int
Email: Quirien.Wijnands@esa.int

ABSTRACT

The verification and validation of a component, algorithm, technology or related standards is a key process in the lifecycle of space applications and products. Classical TestBenches are designed for supporting verification and validation activities where a dedicated infrastructure provides the specific simulation environment and mission context. Due to the different phases during the mission development (starting from phase 0 up to the final product delivery) several TestBenches shall cope with different validation tasks and the support required for them may focus on different environment/characteristics/fidelity (e.g. real time, OBSW in the loop, use of MIL1553 bus, use of SCOS etc). A single infrastructure that optimizes the verification and validation effort through different configurations, tasks and mission needs is a powerful support in verification and validation activities.

In the context of the ATB-RAC (Avionic TestBench – Requirements and Architecture Consolidation) activity the concept of an *evolving* architecture has been defined and applied to an End to End Avionics System TestBench (E2E-ATB) infrastructure. This infrastructure is the evolution of the Avionics Test Bench developed over the past years at ESTEC laboratory, capable of offering the necessary mission context for validation activities. The use of an *evolving* infrastructure architecture optimizes the modularity, resources deployed and synergy of the validation and verification activities between the infrastructure configurations that offer the mission simulation environment and context. This approach allows a fast and easy mission configuration (i.e. selecting mission type, simulation configuration and test to run) and a strong re-use of components while in classical infrastructures these aspects may lead to significant changes in the architecture.

This paper will focus on the description of the TestBench *evolving* architecture and related benefits of its use in performing the validation and verification activities of the space avionics at different levels. The experience and outcome of defining this kind architecture in the context of the ATB-RAC is reported.

INTRODUCTION

Over the past years the definition of a TestBench has been driven, designed and configured around specific verification and validation tasks and particular simulation configuration. Different infrastructures were developed for avionics validation and verification activities involved in different space missions and developments. The adaptation to different scenarios or the re-use of components between classical TestBenches may lead to a huge integration/modification effort that results in an inefficient process. A single TestBench that offers the user with a easy and quick process for mission instantiation (i.e. selecting mission type, simulation configuration and test to run) and the possibility to re-use components between configurations and missions is a desirable infrastructure and powerful support for verification and validation activities. In this sense a first step in optimization of the validation activities effort through different simulation configurations is the definition of a modular architecture capable of mission instantiation and the re-use of components where automatic regression tests capabilities offer a verification method between configurations.

In the frame of the ATB-RAC activity the concept of *evolving* infrastructure architecture has been defined and adapted to the ESA E2E-ATB infrastructure. This way, the E2E-ATB infrastructure has been designed to support validation and verification activities from the functional level (i.e. GNC algorithm validation) up to real time simulation with use of OBSW in the loop, MIL1553 bus, On-Board Computer, HW in the loop, SCOS, etc.

REQUIREMENTS DEFINITION

The definition of a TestBench capable of validation and verification activities of avionics involved in space missions starts from the identification of the potential use and activities that may be performed with a validation infrastructure.
In the frame of the ATB-RAC activity the missions of ESA interest for validation activities are represented by the following reference mission types:

1. Exploration Mission: one or more fixed crafts (lander and/or rover) on the Mars surface, one or more Mars data relay spacecrafts, one or more Earth data relay spacecrafts and one or more Earth based ground stations.
2. Formation Flying Mission: at least two spacecrafts in formation where the GNC algorithms are capable of centralized constellation control.
3. GEO Mission: a geostationary telecom satellite.
4. Earth Observation Mission: an Earth observation satellite.

The definition of the E2E-ATB architectural design has been carried out according to the following development steps:

- Testbench use cases definition
- Testbench system and software requirements definition
- Testbench architecture definition

Infrastructure Use Cases Definition

The first step in the architectural design is to identify and define the purpose, the use cases and the type of users of the TestBench.
The activities supported by the TestBench are directly or indirectly linked to the implementation of missions (and/or their related technologies). Thus, the consideration of the mission/product engineering phases is important for a comprehensive view of all required tasks and the support required for them.
This initial phase defines clearly the activities and the category of users that will use the TestBench that offers the mission context to perform validation activities.
In the frame of the ATB-RAC the main use cases and users were defined at the beginning of the activity and resumed in the following categories:

- *Standards and technology demonstration*. It includes technology development, demonstration and verification activities for developing and maturing the required technologies for the missions. In particular, On-Board SW (OBSW) and avionics-related applications for space need to comply with all applicable standards (ECSS family). The main users of this category are simulation engineers, Attitude and Orbit Control System (AOCS) engineers, OBSW engineers, On-Board Data Handling (OBDH) engineers and system engineers.
- *Technology assessment in support of projects*. It includes all design, implementation integration and validation activities that can be supported along the different mission phases by the E2E-ATB (e.g. algorithms performance assessment, system functional verification, real-time verification, representative OBDH subsystem, etc). This use case is the projection of the first case towards the full project/mission level. The main users of this category are AOCS engineers, OBSW engineers, OBDH engineers and system engineers.
- *Staff competence related activities*. It includes all use possibilities related to staff training and familiarisation with the design, implementation and verification tasks involved in Spacecraft technologies. The main users of this category are simulation engineers, AOCS engineers, OBSW engineers, OBDH engineers and Electronic Ground Support Equipment (EGSE) engineers.

Infrastructure System And Software Requirements Definition

Based on the use cases definition, a set of system and software requirements are derived for driving the architectural design of the validation infrastructure. At this point the functionalities and the logical models of the TestBench have been defined specifying the components/tools that are part of the validation infrastructure. In order to exhaustively identify the validation infrastructure functionalities, the requirements are defined focusing on:

- Top-down functionalities identification.
- Functionalities description.
- Component identification.
- Interface identifications
- Component/Functionality reuse between configurations

In the frame of the ATB-RAC activity the main functionalities/elements identified for the E2E-ATB validation system are resumed in the following points:

- *Monitoring & Control functions*: that are composed by one or several front-ends that allow the user to execute commands or access to monitored data during simulation.

- *Simulation Facility and Environment*: that provides the simulation environment to simulate the spacecraft/ground/universe behaviours which cannot be represented by the rest of the components that take part in the Test Facility
- *Visualization functions*: in charge of create 3D visualization of the simulation, during run-time and off-line from simulation output files..
- *Instantiation functions*: that extract, link and create all the models/files necessary for a specific ATB configuration, for a specific reference mission and for a specific simulation/test.
- *Database*: that serves as the main repository where all parameters needed for instantiation of configurations and parameterisation of missions are stored and taken under configuration control.
- *Repository:* that serves as a ordinary file server in charge of storing the models/sources for each configuration and mission, storing the scenario files (input and simulation output) and storing the ATB framework documentation.
- *Document & Reporting functions*: that generates the on-line libraries documentation, facility documentation and test reports, as well as performing checking on the consistency between the documentation and/or models.
- *Post-Processing functions*: that provide the user with the capabilities to perform post processing analysis on the scenario stored output data, data transformation to human readable format and close-out comparison
- *Plotting functions*: that provide the user with the capabilities to generate plots of the simulation stored data (scenario outputs) and/or post-processing data
- *Coding Rules checking functions*: that provide the user with the capabilities to define and/or select code rules for checking the source code of the OBSW, models or functions.
- *Autocoding functions*: that allow the user to convert the behavioural models to C language and wrap them as SMP2 [1] compliant components capable of running on real-time test facilities increasing reuse between simulation configurations
- *Requirement Management functions*: that allows the user to verify the fulfilment of the system requirements
- *Regression Testing functions*: that provide the user with the capabilities of performing regression test on the system components/models.
- *Components/models re-use*: extensive re-use of available and validated components/models in the ATB system elements.

ARCHITECTURAL DESIGN

Since the validation infrastructure will be used by different users for supporting different activities and missions, the architectural design must support easily instantiation for 4 different configurations (verification facilities) for the specific mission and test case according to the following high level design approach for each configuration:

1. The Functional Engineering Simulator (FES, Fig. 1) allowing the verification of critical elements of a baseline system design (such as Data Handling and AOCS/GNC algorithms) and to support the analysis of system performance.

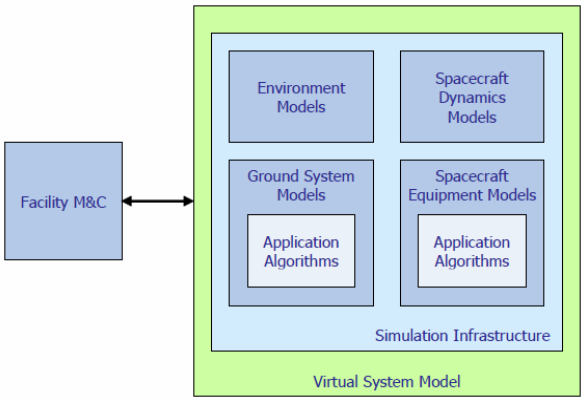

Fig. 1. High level design approach for FES configuration (Behavioural simulation infrastructure) [2]

2. The Functional Validation Test-bench (FVT, Fig. 2) allowing the simulation of a system running under real-time environment, with a focus on the identified critical and prototyped/bread-boarded elements.

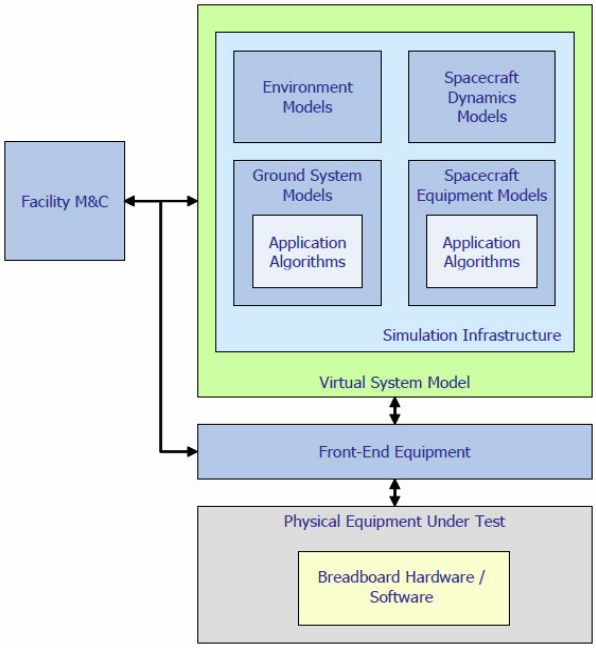

Fig. 2. High level design approach for FVT configuration (Real Time infrastructure) [2]

3. The Software Validation Facility (SVF, Fig. 3) allowing the validation of the on-board software, which needs to be performed in a context that is representative of the spacecraft system in space and with ground interfaces. The SVF contains a fully functional and performance representative simulation model of the spacecraft hardware (e.g. On-Board Processor emulator) where the on-board software is validated.

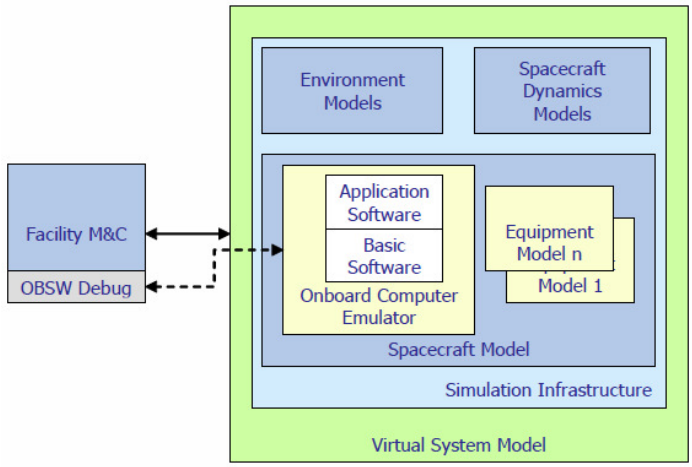

Fig. 3. High level design approach for SVF configuration (OBSW in the loop) [2]

4. The Real-Time Test Bench (RTB, Fig. 4) allowing the validation activities that involve HW in the loop configurations (for at least the OBC to provide full representative real time conditions) with full real interfaces for TM/TC and commanding (EGSE).

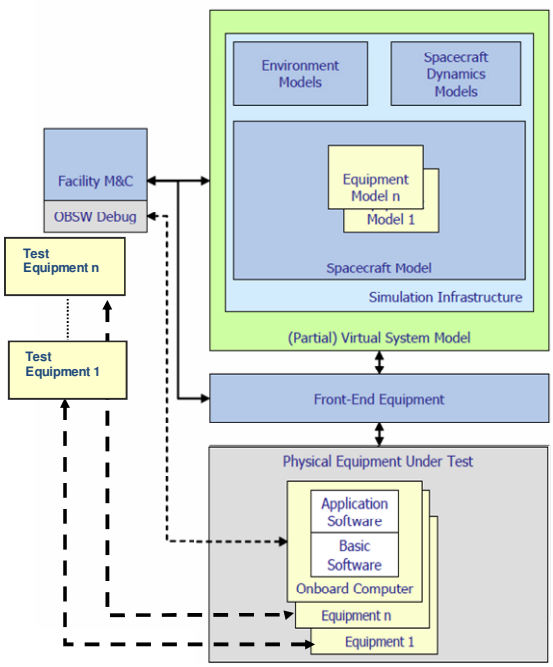

Fig. 4. High level design approach for RTB configuration (at least OBC in the loop) [2]

The validation infrastructure architectural design is performed according to the following key drivers:

- *Modularity*
  The system elements shall be identified and designed as modules that interact with the system. Common elements to the configurations (FES, FVT, SVF and RTB) shall be identified and the next configurations can be obtained adding components typical from the next configuration to the previous one optimizing the evolution of the system through high level configurations. The modularity allows also replacing elements of the same configuration with more detailed models or even HW models.
- *Plug & play*
  The infrastructure shall envisage the re-use of models from different projects in the different facilities in an easy and fast way.
- *Configurability*
  The infrastructure is identified by different number of facilities (FES, FVT, SVF and RTB), configurations (identifying the mission elements) and versions. Each instantiation requires a number of software sources, models and initialization parameters that are stored in repositories and in a database.
- *Autocoding*
  The reuse of models and the portability from the FES configuration up to the RTB is an important aspect of the validation system. This aspect will be optimized by the use of autocoding techniques and tools that guarantee the portability from functional models to C code (i.e. SMP2) allowing incremental step-wise validations on consecutive configurations.
- *Traceability*
  The system components and infrastructure shall be compliant with a set of requirements that shall be verified.
- *Maintainability*
  The validation infrastructure can continuously evolve adding to the system new models, subsystem or OBSW elements to verify and validate in the frame of future activities or missions. Focusing on this aspect the system shall clearly identify a structure for all its elements using a repository for the models, repository for the documentation and a repository for the configuration software in order to optimize the effort for the system maintainability.
- *Regression tests capabilities*
  The models developed in the FES configuration are then ported to C code and used in the other configurations (FVT, SVF and RTB). The infrastructure shall provide automatic regression tests capabilities to compare the model behaviour from non real time environment to real-time saving the effort for a new validation of the functional behaviour of the models.

In the frame of the ATB-RAC activity, the term *evolving* refers to an architectural design of the validation infrastructure capable of being configured in one of the four above configurations where the validation activities may be based on previous configurations in the evolution FES→FVT→SVF→RTB. In this sense the FES configuration is used and embedded by the FVT configuration; the FVT is used to build the SVF but not fully embedded since the Front-End Equipment (FEE) and Physical Equipment/Product Under Test (PUT) are not used in SVF; lately, the RTB configuration uses both FVT and SVF to be built, since it includes OBSW (from SVF configuration) with HW in the Loop (at least OBC but could be also other type of HW, as already present in the FVT). These four different configurations are intended to be used in an incremental manner although the evolution and building of each one of them is not necessary sequentially based on the previous one. The use of the *evolving* infrastructure can start directly at any point of the evolution chain and can be instantiated for a reference mission in a particular configuration.

E2E-ATB EVOLVING ARCHITECTURAL DESIGN APPROACH

The functional diagram (Fig. 5) shows the most representative software and hardware elements of the baseline approach for the architectural design of the E2E-ATB. It can be noticed that any configuration is built over the previous configuration (FES→FVT→SVF→RTB) by integrating the new configuration delta-elements and allowing a new step in the validation process.

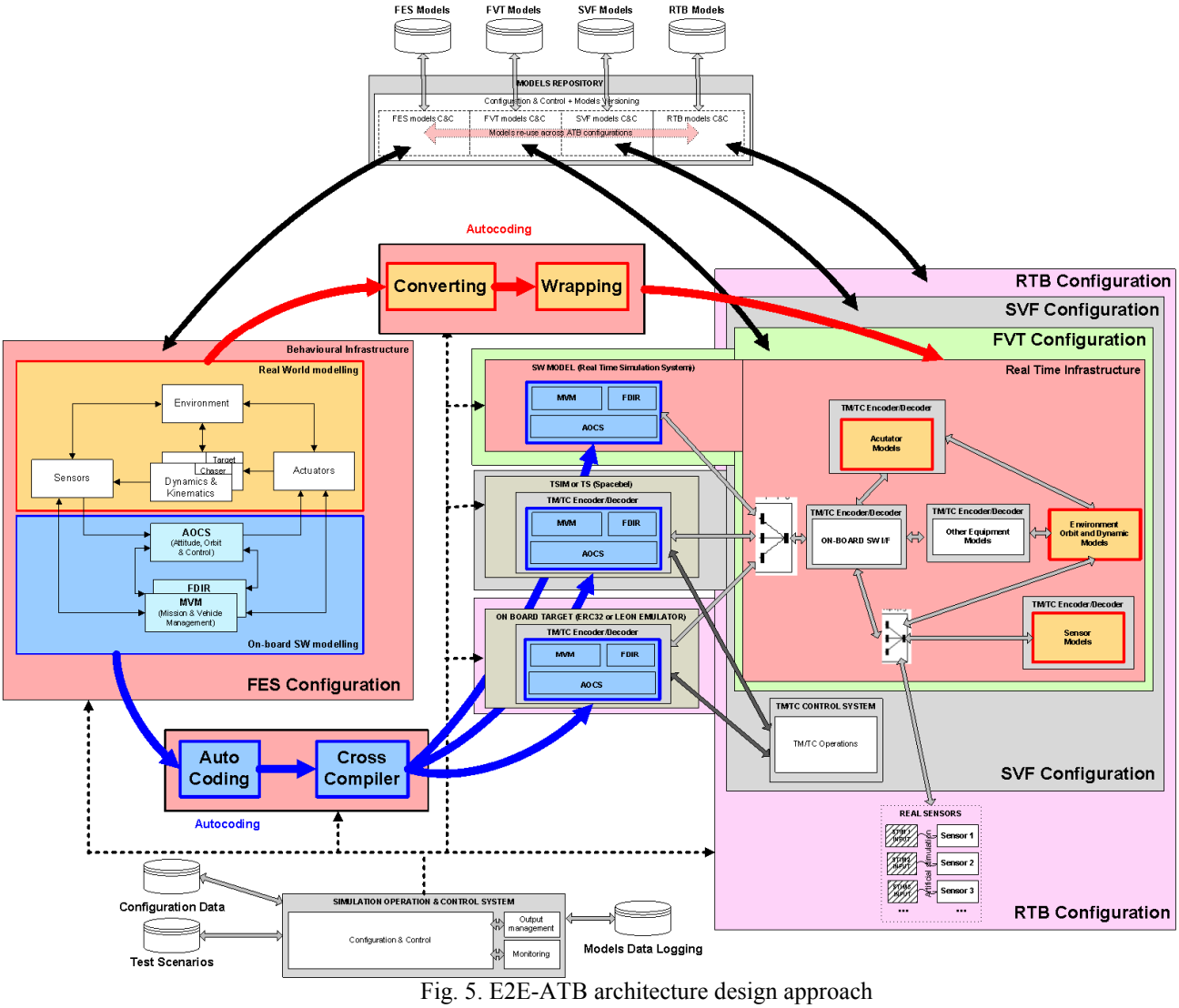

Fig. 5. E2E-ATB architecture design approach

Additionally, it can be observed in Fig. 5 that specific modules/models repositories for each ATB configuration allow storing and later re-using of such modules/models. These repositories shall be used not only when going from a lower

ATB configuration to an upper ATB configuration (e.g. FES→FVT) but also if the ATB user wants to build a specific intermediate ATB configuration (e.g. SVF) without passing though (i.e. without building) the corresponding FES and FVT configurations. The ATB allows such direct intermediate ATB configuration building assumed the models repositories are adequately populated.

The *evolving* infrastructure concept includes *modularity*, *configurability*, *maintainability, portability* and *automatic test capabilities* as a major architectural design driver. This modularity, configurability, maintainability and portability refer to the different verification configurations (FES→FVT→SVF→RTB) where the evolution from a configuration to another is performed by incremental steps. Consecutive validations and verifications activities performed along the sequential configurations will maximize the involved synergies adopted for these tasks. One of the main objectives of the *evolving* validation infrastructure system is to increase the modularity and configurability in order to provide to the user the capability to instantiate a specific configuration (FES, FVT, SVF and RTB) for a specific mission and consequently generate a testing environment on which the user can execute specific analysis and tests.

Architecture Design: Modularity

Modularity is related to the easy and clean capability to substitute an already existing model by a new one with only adaptations at I/F level. The following modularity aspects are included in the design of the E2E-ATB *evolving* infrastructure:

- At the very high level, a clear separation between the OBSW models and the environment models (Real World) is performed.
- Internally in the OBSW model, a clear and clean separation between the different modelled subsystems is kept. In addition, with such modular architecture and modelling approach, it is possible to integrate the OBSW models in several steps and allow incremental verification and performance assessment.
- Internally to the simulation models, the simulation variables are multiplexed, vectorized and demultiplexed between each couple of components.
- A dedicated input file containing all the data settings is used to initialize each one of the simulation blocks. If a block is added, changed or removed, only the single input data file shall be updated.
- Repeated models (mathematical operations, transformation matrices, ...) along the simulator are linked to the corresponding library block. If a basic algorithm is changed, it is only needed to change it at library level.

Architecture Design: Configurability

The configuration management of in the E2E-ATB offers the user the possibility to easily configure and adapt the validation infrastructure (through a dedicated Configurator tool) to a specific mission, configuration and test case.

The Configurator tool is supported by a collection of functionalities to automatically extract modules and models from the System Database and Model/Source repository and plug them in the desired configuration instantiation (see Fig. 6).

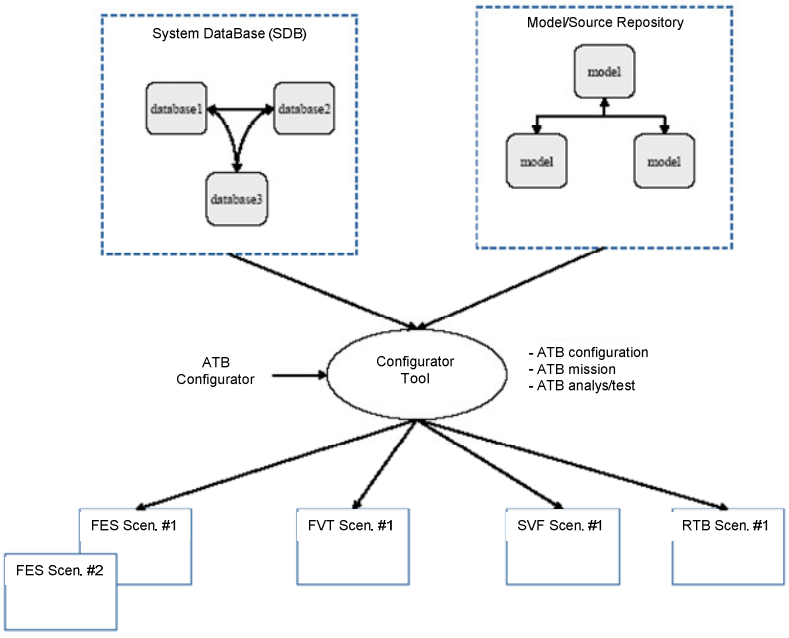

Fig. 6. Configuration approach for the E2E-ATB validation infrastructure

Architecture Design: Maintainability

The E2E-ATB validation infrastructure is designed as a system environment and corresponding components that can continuously evolve adding new models, subsystem or OBSW elements to verify and validate in the frame of future activities or missions. New elements can be part of the configurations at different levels of simulation and different users can have access to the infrastructure resources. The maintainability of the infrastructure is reached by the following points:

- All the elements of the system are identified and organized a structure. This allows the user an easy access and identification of the TestBench elements and functionalities for modification or inclusion of new components.
- The infrastructure models and configuration software are organized in a repository where the user can easily access and configure the elements for the different configurations.
- The system documentation is stored and organized in a repository where the user can easily access it and follow the evolution of the system.
- The system design accounts for elements and functionalities that are as simplest as possible (reduced elements complexity and size).
- The system provides the unit tests of the components and its traceability into the system.
- The development environment information is available into the system (i.e. simulation infrastructure version used to develop the models and algorithms)

Architecture Design: Models Portability

The E2E-ATB validation infrastructure re-uses extensively models from the FES configuration to the real-time configurations by conversion to SMP2 standards through automatic autocoding techniques from functional models.

Architecture Design: Automatic Test Capabilities

The E2E-ATB validation infrastructure performs the automatic regression tests over the models ported from a FES configuration to the real-time configuration saving the effort for a new validation of the functional behaviour of the models developed in the functional simulation environment.

CONCLUSIONS

Usually the validation and verification activities performed in space applications are supported by one or several TestBenches that provides the necessary environment and mission context. A unique validation infrastructure capable of easy and fast mission instantiation for different configurations and the re-use of components as far as possible lead to the concept of *evolving* architectural design.

The use of an *evolving* infrastructure architecture optimizes the modularity, resources deployed and synergy of the validation and verification activities of avionics through different infrastructure configurations (FES→FVT→SVF→RTB) according to the mission needs.

In the context of the ATB-RAC activity a validation infrastructure was designed adopting the concept of *evolving* architecture where the validation activities may be based on previous configurations in the evolution FES→FVT→SVF→RTB but not necessarily starting from the beginning of the chain. The E2E-ATB infrastructure was designed focusing on the following drivers:

- Maximization of the synergy between the configurations (FES→FVT→SVF→RTB) in terms of software reuse and configurations
- Configurability of the infrastructure to allow easy configuration and deployment of a certain configuration for a specific reference mission and for a specific configuration
- Modularity and maintainability to allow easy modification and extensions to solve mission specific problems and to support a plug-and-play approach
- Use and migration of the simulation models to the SMP2 specification
- Automatic testing capabilities

The *evolving* architecture design approach offers the users the possibility to easily expand and adapt the validation infrastructure for future missions validation activities using a unique infrastructure.

REFERENCES

[1] Simulation model portability 2.0 handbook, EGOS-SIM-GEN-TN-0099, 1.2, 28/10/2005

[2] ETM-10-21 Space Engineering - System Modelling and Simulation, ETM-10-21, 2006